CLAIMS

1. Method of making secure the execution of a computer program (EXE) including a set of at least one instruction, which method is characterized in that it includes:

- a first step (E30), prior to the execution of said computer program, of calculating and storing a first signature (SIG1) representative of the intended execution of said set of instructions,

- a second step (E50), during the execution of said set of instructions, of calculating and storing a second signature (SIG2) representative of the execution of said set of instructions, and

- a step (E60) of detecting an anomaly in the execution of said set of instructions on the basis of said first signature (SIG1) and said second signature (SIG2).

2. Method according to claim 1, characterized in that said first calculation and storage step (E30) is executed during the generation of the instructions (A1, A13) of said computer program.

3. Method according to claim 1 or claim 2, characterized in that said second signature (SIG2) stored during said second calculation and storage step (E50) is retained in memory during the execution of at least one second instruction following said set of instructions.

4. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) is obtained from the number of instructions in said set of instructions,

- the second signature (SIG2) is obtained from the number of instructions from said set of instructions that have been executed, and in that

- the detection step (E60) detects an execution anomaly when the first signature (SIG1) and the second

signature (SIG2) are different after the execution of said set of instructions.

5. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) is obtained from the number of instructions in said set of instructions,

- the second signature (SIG2) is obtained from the number of instructions from said set of instructions that have not been executed, this second signature (SIG2) being calculated from said first signature (SIG1), and in that

- the detection step (E60) detects an execution anomaly when the value of the second signature (SIG2) is not zero after the execution of said set of instructions.

6. Method according to claim 5, characterized in that an interrupt of said computer program is triggered when the value of said second signature (SIG2) is below a predetermined threshold.

7. Method according to claim 5 or claim 6, characterized in that the first signature (SIG1) and the second signature (SIG2) are retained in memory during the execution of said program in the same register (REG1).

8. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) is obtained from the code of a critical instruction of said set of instructions,

- the second signature is obtained from the code of said critical instruction, that code being stored at the same time as or after the execution of said critical instruction, and in that

- the detection step (E60) detects an execution anomaly When the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

9. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) is obtained from the address of a critical instruction of said set of instructions, said address being obtained during or after the generation of the executable code of the set of instructions,

- the second signature (SIG2) is obtained from the address of said critical instruction, that address being stored (E30) at the same time as or after the execution (E30) of said critical instruction, and

- the detection step (E60) detects an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

10. Method according to claim 8 or claim 9, characterized in that said critical instruction is a jump instruction (JMP, JNZ, CJNE, JZ).

11. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) and the second signature (SIG2) are error detector codes (CRC1, CRC2) calculated from the code or from an address of an instruction of said set of instructions, and in that

- the detection step (E60) detects an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

12. Method according to claim 11, characterized in that said error detector codes (CRC1, CRC2) are cyclic redundancy check codes.

13. Method according to claim 11, characterized in that said error detector codes are obtained by the logical combination (XOR) of the code or an address of at least one instruction of said set of instructions.

14. Method according to any one of claims 1 to 3, characterized in that:

- the first signature (SIG1) and the second signature (SIG2) are respectively obtained during the generation and the execution of said instructions from at least two elements chosen from:

. the number of instructions in said set of instructions,

. the code of at least one instruction of said set of instructions,

. the address of at least one instruction of said set of instructions, and

. an error detector code calculated from the code or an address of at least one critical instruction of said set of instructions, said address being obtained during or after the generation of the executable code of the set of instructions, and in that

- the detection step (E60) detects an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

15. Method according to any one of claims 1 to 14, characterized in that it includes a step (E70) of destroying at least a portion of the system on which said computer program is executed, this step of destroying being made when an execution anomaly is detected in said detection step.

16. Method according to any one of claims 1 to 15, characterized in that said first signature (SIG1) is generated automatically (E30).

17. Device for processing a computer program including a set of at least one instruction, characterized in that it includes means (12) for calculating and storing the first signature (SIG1) representative of the intended execution of said set of instructions prior to the execution thereof.

18. Device according to claim 17, characterized in

that said means (12) for calculating and storing said first signature (SIG1) are adapted to calculate and store information obtained from the number of instructions of said set of instructions.

19. Device according to claim 17, characterized in that said means (12) for calculating and storing said first signature (SIG1) are adapted to obtain and store information obtained from the code of a critical instruction of said set of instructions.

20. Device according to any one of claims 17 to 19, characterized in that it further includes means (14) for generating executable code from said computer program (SOURCE).

21. Device according to claim 20, characterized in that said means for calculating and storing said first signature (SIG1) are adapted to obtain and store information obtained from the address of a critical instruction, said information being obtained of said set of instructions by said means (14) for generating executable code.

22. Device according to claim 19 or claim 21, characterized in that said critical instruction is a jump instruction (JMP, JNZ, CJNE, JZ).

23. Device according to claim 17, characterized in that said means (12) for calculating and storing said first signature (SIG1) are adapted to calculate and store information obtained from an error detector code (CRC1) calculated from the code or an address of at least one instruction of said set of instructions.

24. Device according to claim 23, characterized in that said error detector code (CRC1) is a cyclic redundancy check code.

25. Device according to claim 23, characterized in that said error detector code is obtained by a logical combination (XOR) of the code or an address of at least one

instruction of said set of instructions.

26. Device for making secure the execution of a computer program including a set of instructions comprising at least one instruction, which device is characterized in that it includes:

- a first register (REG1) for storing a first signature (SIG1) representative of the intended execution of said set of instructions,

- means (22) for calculating and storing in a second storage register (REG2) during the execution of said set of instructions a second signature (SIG2) representative of the execution of said set of instructions, and

- means (24) for detecting an anomaly in the execution of said set of instructions on the basis of said first signature (SIG1) and said second signature (SIG2).

27. Device according to claim 26, characterized in that said calculation and storage means are adapted to retain said second signature (SIG2) in the second register (REG2) during the execution of at least one second instruction following said set of instructions.

28. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) being obtained from the number of instructions of said set of instructions, said second signature (SIG2) calculated and stored by said calculation and storage means (22) is obtained from the number of instructions of said set of instructions that have been executed and in that said detection means (24) detect an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

29. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) being obtained from the number of instructions of said set of instructions, said second signature (SIG2) calculated and

stored by said calculation and storage means (22) is obtained from the number of instructions of said set of instructions that have not been executed, this second signature (SIG2) being calculated from said first signature (SIG1), and said in that detection means (24) detect an execution anomaly when the value of second signature (SIG2) is not zero after the execution of said set of instructions.

30. Device according to claim 29, characterized in that it further includes means for triggering an interrupt of said computer program when the value of said second signature (SIG2) is below a predetermined threshold.

31. Device according to claim 29 or claim 30, characterized in that the first signature (SIG1) and the second signature (SIG2) are stored in said first register (REG1) during the execution of said program.

32. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) being obtained from the code of a critical instruction of said set of instructions, said second signature (SIG2) calculated and stored by said calculation and storage means (22) is obtained from the code of said critical instruction, the code being stored at the same time as or after the execution of said critical instruction, and in that said detection means (24) detect an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

33. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) being obtained from the address of a critical instruction of said set of instructions, said second signature (SIG2) calculated and stored by said calculation and storage means (22) is obtained from the address of said critical instruction, that address being stored at the same time as

or after the execution of said critical instruction, and in that said detection means detect an execution anomaly when the first and second signatures are different after the execution of said set of instructions.

34. Device according to claim 32 or claim 33, characterized in that said critical instruction is a jump instruction (JMP, JNZ, CJNE, JZ).

35. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) and said second signature (SIG2) being error detector codes (CRC1, CRC2) calculated from the code of an instruction of said set of instructions, said detection means (24) detect an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

36. Device according to claim 35, characterized in that said error detector codes (CRC1, CRC2) are cyclic redundancy check codes.

37. Device according to claim 35, characterized in that said error detector codes are obtained by a logical combination (XOR) of the code or an address of at least one instruction of said set of instructions.

38. Device according to claim 26 or claim 27, characterized in that, said first signature (SIG1) being obtained from at least two elements chosen from:

- the number of instructions of said set of instructions,
- the code of at least one instruction of said set of instructions,
- the address of at least one instruction of said set of instructions, and
- an error detector code calculated from the code or the address of at least one instruction of said set of instructions,

said second signature (SIG2) calculated and stored

by said calculation and storage means (22) is obtained in a similar manner from said at least two elements during the execution of said instructions and in that said detection means (24) detect an execution anomaly when the first signature (SIG1) and the second signature (SIG2) are different after the execution of said set of instructions.

39. Device according to any one of claims 26 to 38, characterized in that it further includes means for destroying at least a portion of said computer program.

40. Microcircuit card characterized in that it includes a securing device (100) according to any one of claims 26 to 39.